Collision-free Path Planning for Remote Laser Welding

András Kovács
Fraunhofer Project Center for Production Management and Informatics,
Computer and Automation Research Institute, Budapest, Hungary
andras.kovacs@sztaki.mta.hu

Abstract

The paper proposes algorithms for collision-free path planning in robotic Remote Laser Welding (RLW), using collision detection on a triangle mesh representation of the moving objects and a path planning algorithm based on a classical A* search, both highly specialized to the needs of RLW. The algorithms depart from an optimized task sequence and an initial, potentially colliding rough-cut path. The algorithms modify this path to eliminate all collisions while preserving the stitch sequence and minimizing the cycle time. The approach is validated in computational experiments on real industrial data involving the welding of a car front door.

Introduction

A recent technological trend in the assembly of sheet metal parts, such as car bodies, is the spreading application of Remote Laser Welding (RLW). This contactless technology eliminates the most important limitation of earlier joining techniques, the accessibility issues between the welding gun and the workpiece, by welding from a distant point using a laser beam emitted from a laser scanner that is moved by an industrial robot. This results in up to 80% lower cycle times, reduced operating costs, and higher freedom in part design (Park and Choi 2010). For conventional machining technologies, on-line programming by manual guidance of the robot is the typical programming approach. However, due to the redundancy in the degrees of freedom of the RLW robot and the laser scanner, on-line programming is hardly possible for RLW. However, efficient off-line programming methods tailored to the needs of RLW hardly exist (Reinhart, Munzert, and Vogl 2008).

Our general objective is the development of an interactive off-line programming toolbox with efficient optimization capabilities for RLW (Erdős et al. 2013). In a recent paper (Kovács 2013), we have introduced an efficient algorithm for integrated task sequencing and rough-cut path planning. Significant novelties of the algorithm include (1) the explicit modeling of the coupled movement of the quick tool (laser beam repositioning) and the relatively slow robot; (2) exploiting the high degree of freedom in choosing the robot path when welding a well-defined stitch position; and finally, (3) planning in the continuous space, without losses stemming from sampling that characterizes many other approaches working on a discretized space representation. Nevertheless, that algorithm ignores potential collisions along the path. This assumption is not absolutely unrealistic, since general (fixture) design guidelines for RLW require that the access volumes of the welding stitches are left clear to preclude collisions. However, our experience on real industrial data showed that this requirement is sometimes overridden by other design objectives, and collisions do occur on the computed rough-cut path.

In this paper, we propose a collision-free path planning algorithm that departs from the above rough-cut path, and modifies it to avoid any collisions while preserving the original stitch sequence and minimizing the cycle time. We present collision detection techniques on triangle mesh models and a path planning algorithm based on classical A* search, both highly specialized to the needs of RLW. These include searching for a trajectory that visits given regions of the 3D space in a pre-defined order and spends the time required to execute the corresponding actions in each of those regions. The definition of collision depends on the action executed in a given position. A trajectory that minimizes the cycle time is looked for.

The paper is organized as follows. First, a brief review of the related literature and the technological background is given. Then, the path planning problem is defined formally. Afterwards, the proposed collision detection and path planning methods are presented in detail. Finally, computational experiments are reported and conclusions are drawn.

Literature Review

The RLW technology, including its benefits and limitations, is presented in (Tsoukantas et al. 2007). Applications of RLW in the automotive industry are reviewed in (Shibata 2008). The importance of automated process planning for RLW is emphasized in (Hatwig, Reinhart, and Zaeh 2010).

We are aware of a single earlier approach to task sequencing and path planning specifically for RLW and remote laser cutting, introduced in a series of papers (Reinhart, Munzert, and Vogl 2008; Hatwig et al. 2012). The proposed algorithms are designed mostly for planar workpieces: task sequencing is performed by solving a traveling salesman problem (TSP) over the fixed welding stitch positions, and a robot path is computed in a plane above the workpiece. Potential collisions are ignored. A similar model is applied

and construction heuristics are proposed for path planning in laser cutting in (Dewil, Vansteenwegen, and Cattrysse 2014). The applied model also captures sophisticated ordering constraints among the contours to be cut.

An efficient, generic task sequencing and collision-free path planning model, with illustrations from resistance spot welding (RSW) is presented in (Saha et al. 2006). A critical assumption is that the robot can execute each effective task from a relatively small set of candidate configurations, e.g., at most 10 configurations per task, which can be generated a priori. An iterative algorithm is proposed that tries to compute as few point-to-point collision-free paths as possible, hence avoids solving unnecessary computationally demanding subproblems. The difficulty in applying this approach to RLW stems from the fact that efficient paths in RLW exploit the free movement of the robot in the continuous space while welding.

The minimization of processing time in a milling operation is investigated in (Castelino, D'Souza, and Wright 2002). A Generalized TSP (GTSP) approach is proposed, where the nodes correspond to the candidate tool entry/exit points for machining a feature. Potential collisions are neglected. A TSP with Neighborhoods (TSPN) model is proposed in (Alatartsev et al. 2013) for sequencing a set of robotic tasks whose start/end points can be chosen arbitrarily along open or closed contours, such as in the case of different cutting problems. In (Alatartsev, Augustine, and Ortmeier 2013) a construction heuristic, the so-called constricting insertion heuristic is introduced for the derived TSPN over a set of 2D polygons. A multi-objective constraint optimization model is proposed in (Kolakowska, Smith, and Kristiansen 2014) for task sequencing in spray painting, for minimizing cycle time and maximizing paint quality at the same time.

Classical AI methods for collision-free path planning search a discretized grid representation of the environment using algorithms like A* or one of its numerous descendants. These include D*-Lite (Koenig and Likhachev 2005) or Focussed D* (Stentz 1995) for dynamically changing environments, or ARA* (Likhachev, Gordon, and Thrun 2003), an anytime algorithm with provable bounds on sub-optimality. Field D* (Ferguson and Stentz 2006) lifts the constraint on the previous algorithms that they must move through a series of neighboring grid points, thus saving unnecessary turnings and further reducing path length. These methods are suitable mostly for lower dimensional problems due to the computational effort required.

Higher dimensional problems, such as path planning for a robot with many degrees of freedom, are often intractable using the above methods. In such cases, incomplete methods that apply randomization are preferred. The most efficient approaches are *Rapidly-exploring Random Trees* (RRT) (Kuffner and LaValle 2000) for the single-query case, and *Probabilistic Roadmaps* (PRM) (Kavraki et al. 1996; Geraerts and Overmars 2002) for the multiple-query case. A recent tendency is to delegate motion planning to GPUs, see, e.g., (Park et al. 2013), where a highly parallelized RRT algorithm is proposed to exploit the computation capabilities of GPUs.

Path planning algorithms typically rely on external software libraries for collision detection. Such libraries contain, e.g., the Proximity Query Package (PQP) (Larsen et al. 2000) for rigid objects represented as triangle mesh, or V-Collide (Hudson et al. 1997) specifically for VRML applications. A benchmarking suite for pairwise static collision detection algorithms and a comparison of numerous freely available collision detection algorithms has been presented in (Trenkel, Weller, and Zachmann 2007).

The integration of task and motion planning has received significant attention in the robotics community, especially in navigation and manipulation applications. A plethora of approaches has been offered to combine symbolic planners as high-level solvers and motion planners (e.g., PRM or RRT planners) as subproblem solvers, see, e.g., (Kaelbling and Lozano-Perez 2011; Srivastava et al. 2014).

Technological Background

The Welding Process

The recent development of a new generation of laser sources, such as fiber lasers, enabled laser welding with an operating distance (focal length) above one meter. The new technology, RLW, joins sheet metal parts without physical contact or even a close approach. This, on the one hand, ensures extremely fast positioning speed compared to classical RSW, where a vast welding gun must contact the workpiece. The high productivity of the technology results in up to 80% lower cycle times and reduced operating costs, making RLW economically profitable despite the high initial investments. In addition to the direct economic gain, the abolishment of the accessibility issues removes many earlier constraints on part designs, an advantage that can be turned easily into parts with reduced weight, yet higher stiffness. This, in the automotive industry, facilitates the design of lighter and more efficient cars, without compromising safety.

An RLW operation consists in joining two or more sheet metal parts at various joints. In this paper, we assume stitch welds, i.e., linear welding stitches with a typical length of 15-30 mm each. During the operation, the parts are held in a fixture. It is assumed that the operation is performed by a single RLW robot. A typical RLW robot consists of a robot arm with 4 rotational joints and a laser scanner. The robot arm moves the scanner with a maximum speed of 0.2-0.6 m/s, and due to the low scanner weight, with a rather high acceleration. The scanner contains two tilting mirrors for the rapid positioning of the laser beam (up to 5 m/s), and a lens system to regulate the focal length. Hence, the typical RLW robot is a redundant kinematic chain with 7 degrees of freedom, in which the mirrors in the scanner position the laser beam an order of magnitude faster than the movement of the mechanical joints of the robot arm. The process of welding a car door by an RLW robot is depicted in Figure 1.

The robot can weld a stitch if the scanner is located within the *focus range* (e.g., 800-1200 mm) from the stitch, and the *inclination angle* (i.e., the angle between the laser beam and the surface normal) is not more than a specified technological parameter (e.g., 15°). These constraints define a truncated cone above the stitch, which will be called the *tech-*

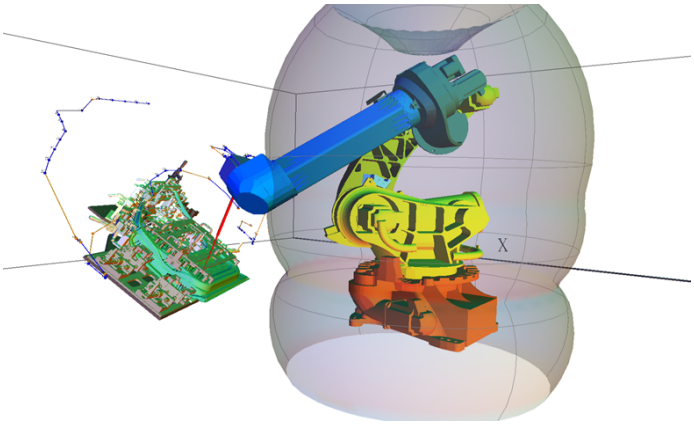

Figure 1: RLW robot welding a car front door, positioned in a fixture. The blue sections of the indicated scanner path represent the movement of the robot while welding, while yellow sections denote idle movement.

nological access volume (TAV) of the stitch, as shown in Figure 2. Strictly speaking, the above definition would require spherical outer and inner bases for the truncated cone. However, to benefit from convex TAVs, we approximate this shape by using a planar inner base, while leaving a spherical outer base. The *collision-free access volume* is the subset of the TAV from which welding can be performed without collisions. Since the length of a stitch is significantly smaller than other characteristic dimensions in the welding process, it is reasonable to assume that all points of a stitch can be processed from the access volume belonging to the mid-point of the stitch.

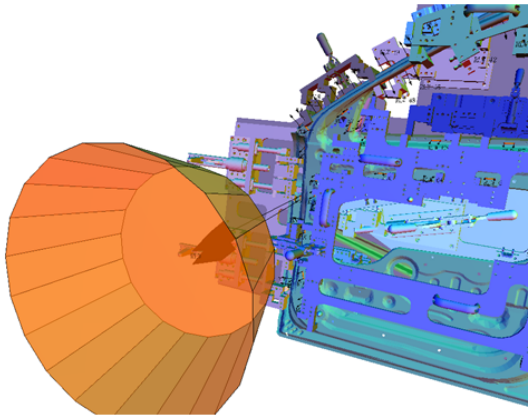

Figure 2: Technological access volume (TAV) of a welding stitch.

Each stitch can be welded at a given speed (e.g., 50 mm/s), which depends on the thickness and the material of the parts to join. Each stitch must be processed without interruption. The robot can weld the stitch while in motion, therefore the trajectory of the scanner must be a curve in the 3D space, such that sufficient time is spent in the access volume of each stitch. There are 30-75 stitches to weld in an RLW operation in the automotive industry.

An Off-line Robot Programming Approach

In industrial practice, robot programming is still typically performed by on-line programming, i.e., by manually guiding the robot from one position to the next, at very small steps, which is a extremely time consuming and hardly feasible for RLW. Our goal is to implement a complete off-line programming toolbox for RLW, which can provide an automated method for computing close-to-optimal robot programs. This involves the optimization of the task sequence, integrated with rough-cut path planning; collision-free path planning in the workpiece coordinate system; the placement of the workpiece in the welding cell; the inverse kinematic transformation that converts the path into the robot joint coordinate system; and finally, the simulation of complete process plan and the automated generation of the robot program code (see Figure 3). The workflow has been presented in detail in (Erdős et al. 2013).

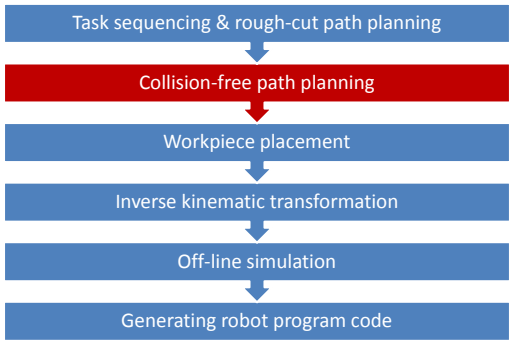

Figure 3: Workflow in the off-line programming system. The paper focuses on the second step, collision-free path planning.

An important consequence of the above workflow is that path planning is performed in the 3D Cartesian coordinate system of the workpiece. This was motivated by the fact that the extensive geometric computations required for the optimization of the task sequence and the robot path cannot be executed efficiently in the robot joint coordinate system (Kucuk and Bingul 2006).

In the recent conference paper (Kovács 2013) we have presented an efficient algorithm for integrated task sequencing and rough-cut path planning, using a detailed technological model of the RLW process. However, that algorithm ignores potential collisions along the path, exploiting that RLW is less exposed to accessibility issues than any other welding technology, and hence, the optimal task sequence is hardly affected by collisions. The objective of the path planning algorithm investigated in this paper is eliminating all collisions from the rough-cut path while preserving the given task sequence and minimizing cycle time.

Problem Definition

The collision-free path planning problem consists in computing a scanner trajectory in the 3D Cartesian coordinate

system attached to the workpiece, such that the robot welds all stitches along the path and the cycle time is minimized. Formally, there is a list of n welding stitches, denoted by $(s_1, s_2, ..., s_n)$, to be welded by an RLW robot in this predefined order, originally computed by some task sequencing algorithm. Each stitch is characterized by its technological access volume, TAV_i, a truncated cone as defined above, a collision-free access volume, $\text{CFAV}_i \in \text{TAV}_i$, and the associated welding time, t_i. Each stitch s_i must be welded without interruption, during which the movement of the scanner is constrained to CFAV_i. Only one stitch can be welded at a time. The path may contain idle robot movement, i.e., sections without welding. Such sections of the path must be located within CF_0, the region that is free of collisions of the robot (with the laser beam switched off).

It is assumed that the maximum robot speed (speed of the scanner), v, is independent of the position in the working area, and the robot has an infinite working area. Finally, the objective is minimizing the cycle time, i.e., the total time required for the robot to travel along the computed trajectory.

Path planning must avoid all types of collisions that can be detected at this phase of the workflow, i.e., that are independent of decisions made in later phases (see Figure 3 earlier). These are the collisions between the *laser beam* vs. the *workpiece* and the *fixture*, as well as the *scanner head* vs. the *workpiece* and the *fixture*. It is noted that these are the most critical types collisions in RLW.

In addition, we assume that there is given an initial, potentially colliding rough-cut path, which has been originally computed by an external algorithm, practically, the earlier proposed task sequencing and rough-cut path planning algorithm. This initial trajectory welds each stitch from TAV_i, but potentially from outside CFAV_i. Below, we propose a procedure that detects collisions along the rough-cut path, and resolves those collisions by a series of modifications to the initial path. The result of applying this method for collision avoidance is shown in Figure 4.

Collision detection

Collision detection is performed using PQP (Larsen et al. 2000) on a triangle mesh representation of the involved 3D objects. The mesh representation of the workpiece and the fixture is given as input, in STL file format, whereas the mesh representation of the laser beam and the scanner head is constructed runtime. Out of the various geometric computation functions offered by PQP, collision detection relies on distance computation between pairs of objects. If the computed distance is smaller than a given threshold, then the two objects are declared colliding in a given robot position. Otherwise, the two objects do not collide. If, in a given robot position, none of the relevant pairs of objects collide, then the position itself is non-colliding.

Collision detection must ensure that the required minimum distance between the relevant pairs of objects is maintained while the robot moves along its *continuous path*. To provide this guarantee based on collision checks performed in an appropriately selected, finite set of *discrete positions*, the following method is applied. For each pair of relevant objects, a *lower tolerance* and an *upper tolerance* distance is introduced, denoted by d_l and d_u, respectively, with $d_l < d_u$. Collision checks in the selected positions are performed with a required minimum distance of d_u, which ensures that a minimum distance of d_l is maintained throughout the continuous path.

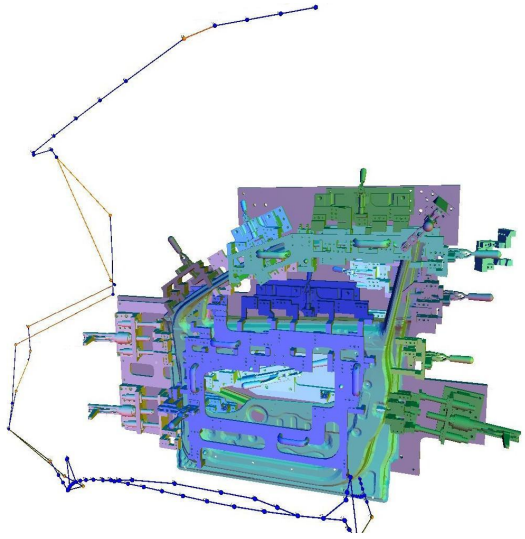

Figure 4: Comparison of the rough-cut and the collision-free paths. Blue sections denote welding, while yellow section correspond to idle movement.

Let us denote by d^* the minimum distance of a given pair of objects along a continuous path. If $d^* < d_l$, then the above method classifies the path as colliding. If $d^* \geq d_u$, then the path is classified as non-colliding. However, if $d_l \leq d^* < d_u$, then the classification is undefined. Hence, parameter d_l specifies the minimum distance required between the objects, while d_u can be used to control the trade-off between geometric accuracy and computational efficiency (number of sample points required).

In the implemented collision detection method, separate tolerance parameters have been considered for the laser beam and scanner head, as shown in Table 1. Moreover, contact between the end of the laser beam and the workpiece is operational: this is the physical core of the welding process. Therefore, when performing collision detection between the laser beam and the workpiece, the beam length is truncated by e^L. No truncation is applied for collision detection against the fixture.

Finally, it is assumed that welding can be performed only when the complete stitch is visible from the laser emission point, and therefore, the theoretical possibility is ignored that portions of the stitch might become visible only gradually, as the scanner head moves along its path and welds other portions of the same stitch. This assumption is common in stitch welding (see, e.g., (Hatwig et al. 2012)).

Collision detection for a single robot position

A key procedure for collision-free path planing is collision detection for a given robot position, P. The definition of collision depends on the action performed in the given position:

Parameters for collision detection	
d_l^S	Lower tolerance distance for the scanner head
d_u^S	Upper tolerance distance for the scanner head
d_l^L	Lower tolerance distance for the laser beam
d_u^L	Upper tolerance distance for the laser beam
e^L	Laser beam end truncation
r^S	Radius of the scanner head model
Parameters for collision avoidance	
ϱ	Resolution of the 3D rectangular grid
B	Maximum bypass w.r.t. the original path
N	Neighborhood size for re-planning

Table 1: Parameters for collision detection and for collision avoidance.

when *welding a stitch*, both the scanner head and the laser beam are considered; during *idle movement*, the laser beam is switched off, and hence, only the scanner head is taken into account. The mesh models of the scanner head and the laser beam are constructed as follows:

Scanner head Since path planning precedes inverse kinematics in the proposed workflow, the orientation of the scanner head is unknown at the time of path planning. Hence, instead of a precise geometric model, the circumscribed sphere of the scanner head is used, which corresponds to a pessimistic assumption. Technically, this is achieved by using a mesh model that represents the scanner head as a single point P, and specifying $r^S + d_u^S$ as the distance threshold value in the PQP distance query.

Laser beam The mesh model of the laser beam for welding a *linear stitch* consists of a single triangle, as shown in Figure 5, corresponding to the assumption that the complete stitch is visible from the given robot position. The triangle is defined by the robot position (laser emission point), P, and the stitch start and end points, S_1 and S_2. In order to avoid false positive results near the workpiece, the height of the triangle is truncated by d_u^L when testing against the fixture, and by $d_u^L + e^L$ when testing against the workpiece. In both cases, a distance threshold of d_u^L is applied, resulting in the light gray collision zone for the fixture and the dark gray zone for the workpiece. In case of a *circular stitch* with radius r, the mesh consists of a single line between the laser emitting point and the stitch center point. The line is truncated by $d_u^L + r$ (fixture) or by $d_u^L + e^L + r$ (workpiece), and the distance threshold is set to $d_u^L + r$, resulting in a thin cylindrical volume that must be collision-free.

Collision detection for a continuous section

Collision detection is performed separately for each linear section of the broken line scanner path. Checking the linear section $\overline{P_1P_2}$ starts by collision detection for position $P = P_1$, and continues by checking subsequent discrete points of the section in the direction of P_2. The size of the discrete steps depends on the results of the distance queries, and it is chosen to guarantee that the prescribed lower tolerance

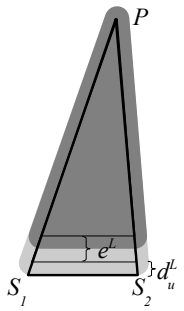

Figure 5: Mesh model of the laser beam for welding the linear stitch $\overline{S_1S_2}$ from robot position P. The approach results in the light gray collision zone for fixture, and the dark gray collision zone for both the fixture and the workpiece.

distance is maintained throughout the continuous path, even at points not directly checked. If all the checked positions are collision-free, then section $\overline{P_1P_2}$ itself is collision-free. Otherwise, the section is colliding. The pseudo-code of the algorithm is presented below.

```
PROCEDURE IsColliding(P1, P2)
    LET P := P1
    LOOP
        LET d := GetDistance(P)
        IF (d < du) THEN
            RETURN TRUE
        ELSE IF P = P2 THEN
            BREAK
        LET s := sqrt(d^2 - dl^2) + sqrt(du^2 - dl^2)
        IF d(P, P2) > s) THEN
            P := P + d(P1, P2) * s / d(P1, P2)
        ELSE
            P := P2
    RETURN FALSE
```

In the pseudo-code, function GetDistance(P) executes a PQP distance query for the single robot position P. The tolerance distance parameters d_l and d_u are set as presented above. The correctness of the procedure is proven in the following lemma, focusing on two subsequent robot positions investigated in the inner loop of the algorithm, denoted as P and P', for collisions of the scanner head, represented as a single point mesh model.

Lemma 1 *Let P and P' be two points in space such that their shortest distance from a given, fixed 3D object O is $d(P, O) = d \geq d_u$ and $d(P', O) \geq d_u$. Now, if $d(P, P') \leq \sqrt{d^2 - d_l^2} + \sqrt{d_u^2 - d_l^2} = s$, then for any point Q of section $\overline{PP'}$, it holds that $d(Q, O) \geq d_l$.*

Proof. Assume that the shortest distance between the object O and the section $\overline{PP'}$ arises between points $R \in O$ and $Q \in \overline{PP'}$ (see Figure 6). If $Q = P$ or $Q = P'$ then the lemma is trivial. Otherwise, Q is an internal point of section $\overline{PP'}$. If $d(P, P') \leq \sqrt{d^2 - d_l^2} + \sqrt{d_u^2 - d_l^2}$, then either $d(P, Q) \leq \sqrt{d^2 - d_l^2}$ or $d(Q, P') \leq \sqrt{d_u^2 - d_l^2}$. Assume that the first case holds. Then, PQR is a right triangle with

hypotenuse d. By the Pythagorean theorem, if $d(P,Q) \leq \sqrt{d^2 - d_l^2}$, then $d(Q,R) \geq l_2$, and the lemma is proven. For the second case, similar claims can be made for the triangle $P'QR$. □

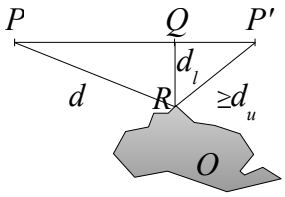

Figure 6: Illustration of the proof of the correctness of the procedure for collision checking on the continuous section $\overline{PP'}$.

It is straightforward to generalize the lemma to the laser beam as well. The proof exploits that the mesh model of the beam consists of triangles with one vertex corresponding to the laser emission point, and two (possibly coinciding) vertices are fixed while the robot moves along its path. Each point of such a triangle moves along a linear section as the laser emission point moves along $\overline{PP'}$.

Collision-free path planning

Representation of the path

It is assumed that the potentially colliding rough-cut path is described as a list $((P_1, a_1), (P_2, a_2), ..., (P_k, a_k))$, where segment (P_i, a_i) denotes that the robot moves from point P_i to point P_{i+1} along a linear section while performing action a_i. Action a_i can be of two types: $a_i = (s_{[i]}, +)$ or $a_i = (s_{[i]}, -)$. Action $a_i = (s_{[i]}, +)$ encodes welding stitch $s_{[i]}$, where $s_{[i]}$ corresponds to one of the stitches $s_1, ..., s_n$, sequenced to the ith position of the path. In contrast, $a_i = (s_{[i]}, -)$ denotes idle movement directly after welding $s_{[i]}$. The rough-cut path contains exactly one segment for welding each stitch, and zero or one idle movement segment between two welding segments, hence, $n \leq k < 2n$. Note that the same does not hold for the collision-free path, since it might be necessary to move the robot along a more complex path to avoid collisions, both while welding and during idle movement.

It is assumed that each segment (P_i, a_i) is labeled as colliding or non-colliding by the above collision detection procedure. Collision avoidance relaxes the colliding segments of the path, as well as the segments that are close to colliding segments. More specifically, segment (P_i, a_i) is relaxed if and only if there exists j such that $i - N \leq j \leq i + N$ and segment (P_j, a_j) is colliding, where N is the neighborhood size for re-planning. The procedure is illustrated in Figure 7, where the colliding segments (red) and their neighborhood with $N = 1$ are re-planned, resulting in a collision-free path (blue).

As a result, the rough-cut path consists of a series of relaxed and non-relaxed segments. Collision avoidance is performed on maximal relaxed sections of the path, and replaces these relaxed sections by new, collision-free sections.

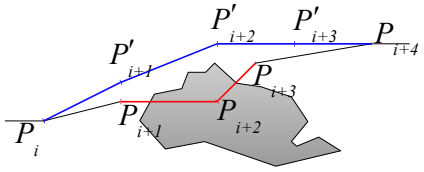

Figure 7: Collision avoidance by replanning the colliding segments (red) and their neighborhood with $N = 1$.

The proposed procedure preserves the order of the stitches, but it may modify the number of segments in the path, as well as the points visited along the path.

In the sequel, we assume that collision avoidance is performed for a single, maximal relaxed section of the rough-cut path, $((P_\alpha, a_\alpha), (P_{\alpha+1}, a_{\alpha+1}), ..., (P_\beta, a_\beta))$. Furthermore, let $(s_{\{1\}}, s_{\{2\}}, ..., s_{\{m\}})$ denote the sequence of welding the stitches along the relaxed path section. If there are several, disjoint relaxed sections to re-plan, then the same procedure is repeated on each of those sections.

Representation of the collision map

The state space for collision-free path planning is represented as a four-dimensional map of discrete vertices, with the three spacial dimensions and one additional dimension describing the action performed in the vertex. The map contains the combination of a 3D point and an action, $(P, a = (s, +))$ as a vertex if and only if P is contained in the CFAV of stitch s. The pair $(P, a = (s, -))$ is contained in the map if P itself is collision-free (with the laser beam switched off), i.e., $P \in \text{CF}_0$.

The points included in the map are the points of a discretized, rectangular 3D grid with a resolution of $\varrho = \min(d_u^S - d_l^S, d_u^L - d_l^L)$. By Lemma 1, the application of this resolution and collision checks in the grid points with tolerance d_u^S and d_u^L ensure that movement between two neighboring grid points is collision-free with d_l^S and d_l^L. The map is created for a finite rectangular area, defined by values $x_{\min}$, $x_{\max}$, $y_{\min}$, $y_{\max}$, $z_{\min}$, and $z_{\max}$, where $x_{\min} = \min_{i=\alpha}^{\beta} x(P_i) - B$ and B is the maximum bypass parameter, and other boundary parameters are computed analogously.

Possible transitions between states are captured by directed arcs between the vertices, according to the following rules. Let $N(P)$ denote the 6-neighborhood of point P, i.e., the set of six neighboring points along the x, y, and z axis. From the vertex capturing action $(s_{\{i\}}, +)$ in P, there are arcs to

- $(s_{\{i\}}, +)$ in $N(P)$, i.e., continuing the welding operation in a neighboring point;
- $(s_{\{i\}}, -)$ in $N(P)$, i.e., finishing the welding operation and continuing with idle movement;
- $(s_{\{i+1\}}, +)$ in $N(P) \cup P$, i.e., continuing with welding the next stitch.

From the vertex encoding $(s_{\{i\}}, -)$ in P, there are arcs to

- $(s_{\{i\}}, -)$ in $N(P)$, i.e., continuing the idle movement in one of the neighboring points;

- $(s_{\{i+1\}}, +)$ in $N(P)$, i.e., welding the next stitch.

To save computation time by omitting unnecessary collision checks, the proposed procedure does not generate the complete collision map at once. Instead, vertices are generated and checked for collisions on the fly, as they are explored by the search procedure. Moreover, the results of collision detection are inferred from the results for the neighboring points whenever possible.

A* search for a collision-free path

In order to compute a collision-free path, an A* search is performed on the above defined collision map. Each node of the search tree is represented as a tuple $\Gamma = (P, a, r, t)$, where P is a 3D point and a is an action, corresponding to a vertex in the collision map. The non-negative real r is the time remaining for welding stitch corresponding to a, and t is the total time of traveling the path from the source node to Γ. Note that if a is an idle movement action, then $r = 0$.

The source node of the search is defined as $(P_\alpha, (s_{\{1\}}, +), t_{\{1\}}, 0)$, and goal states are of the form $(P_\beta, (s_{\{m\}}, +), 0, \cdot)$. A special case arises when the relaxed section is at the beginning of the rough-cut path, since in this case, the collision-free path can start at any point P in $\text{CFAV}_{\{1\}}$. Accordingly, search is initialized with multiple source nodes in the list of open nodes, one for each such point P. Similarly, when the relaxed section is at the end of the rough-cut path, then it can terminate anywhere in $\text{CFAV}_{\{m\}}$.

The cost function of the A* search is t, while the heuristic function h is a lower estimate of the remaining time. In a node $\Gamma = (P, (s_{\{i\}}, \cdot), r, t)$, the heuristic value is computed as

$$h(\Gamma) \;=\; \max\left(r + \sum_{j=i+1}^{m} t_{\{j\}},\; \frac{d(P, P_\beta)}{v}\right).$$

The first term encodes the total remaining welding time on the current stitch and on the future stitches. The second term is the time for traveling from the current location to the goal point P_β. When there are multiple goal points, the second term is ignored.

According to the rules of the A* search, in each step, a node with minimal $t + h$ is expanded. When expanding a node $\Gamma = (P, a, r, t)$, Γ is removed from the open list, and a new node $\Gamma' = (P', a', r', t')$ is created and inserted into the list of open nodes for each directed neighbor of (P, a) in the collision map. The new node inherits P' and a' from the vertex of the map, whereas parameters r and t are computed as follows.

- If a' is welding the same stitch from a different position, then $r' = \max(r - \frac{\varrho}{v}, 0)$ and $t' = t + \frac{\varrho}{v}$;
- If a' is welding the subsequent stitch and $P = P'$, then $r' = t_{\{i+1\}}$ and $t' = t + r$;
- If a' is welding the subsequent stitch and $P \neq P'$, then $r' = t_{\{i+1\}}$ and $t' = t + \max(\frac{\varrho}{v}, r)$;
- If a' is idle movement, then $r' = 0$ and $t' = t + \max(\frac{\varrho}{v}, r)$;

This search step is iterated until a goal state is reached. Links between nodes and their parents are maintained throughout the search, and sequence of links from the first goal state to the source state encodes a collision-free path from P_α to P_β.

Let there be given two search nodes belonging to the same point P, denoted by $\Gamma = (P, a, r, t)$ and $\Gamma' = (P, a', r', t')$. The following two dominance rules are defined.

Dominance rule #1: If $t < t'$ and $t + h(\Gamma) < t' + h(\Gamma')$, then let Γ' be fathomed.

Dominance rule #2: If $t < t'$, then let Γ' be fathomed.

While rule #1 is obviously admissible, the stronger rule #2 is an inadmissible dominance rule, and may result in losing the optimal collision-free path. However, even the application of rule #2 maintains the completeness of the search, i.e., it is guaranteed that a feasible collision-free path is found if there exists one. In our implementation we have decided to apply rule #2, since initial experiments we have found that it brings considerable speed-up with negligible loss of performance.

Smoothing the path

The path computed by the A^* search consists of small, axial sections in the Cartesian coordinate system of the workpiece. This path is smoothed by eliminating the unnecessary breakpoints using an algorithm that considers each section (P_i, a_i) one-by-one. If $a_i \equiv a_{i-1}$ and section $\overline{P_{i-1}P_{i+1}}$ is collision-free for executing action a_i, then this section is removed from the path, which implicitly entails that the previous section (P_{i-1}, a_{i-1}) is extended until point P_{i+1}. The procedure is illustrated in Figure 8. Finally, the smoothed collision-free path segments are inserted at the place of the removed, colliding path segments, and the cycle time is recalculated.

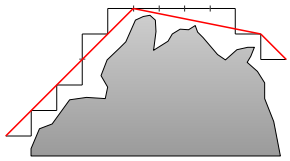

Figure 8: Comparison of the initial (black) and the smoothed (red) collision-free paths.

Experimental Results

Comparison of Different Algorithms

The proposed algorithms have been evaluated on problems involving the assembly of a car front door using RLW. Experiments have been performed on real industrial data, containing a single door geometry with different stitch layouts, various fixture designs, and realistic technological parameters. The instances contained 28-71 welding stitches. The mesh model of the door geometry consisted of ca. 10^5 triangles, while the fixture model contained $5 \cdot 10^5$ triangles.

The experiments involved computing a task sequence and a rough-cut path by three different algorithms for each instance, and converting all the three solutions to a collision-

free path by the algorithm proposed above. The three sequencing algorithms are as follows:

- TS-PP, our algorithm for integrated task sequencing and path planning (Kovács 2013);
- RMV, the single sequencing algorithm dedicated to RLW from the literature (Reinhart, Munzert, and Vogl 2008), which solves a TSP over the stitch positions. Hence, this algorithm focuses on the length of the tool contact point (TCP) path when optimizing the stitch sequence;
- RMV*, a modified version of RMV that solves the TSP over the mid-points of the access volumes, instead of the stitch position. This modification implies that RMV* addresses the minimization of the length of the scanner path, instead of the TCP path.

All algorithms have been implemented in C++. RMV and RMV* used ILOG CP as a TSP solver. The experiments were run on a 2.66 GHz Intel Core 2 Duo computer. A time limit of 120 seconds was applied.

The proposed algorithms computed a feasible, collision-free robot path for every instance with all the three task sequencing methods. The results unambiguously indicate the dominance of robot path planning (TS-PP and RMV*) over TCP path planning approaches, see Figure 9. For workpieces with complex geometry, RMV leads to moving the scanner head in a zigzag above stitches that have nearby positions but different surface normals. In case of a car door, this phenomenon is the most spectacular around the window frame, where the stitches on the inner and the outer sides are close to each other, but must be welded from opposite directions. Consequently, in our experiments, RMV resulted in up to 3 times higher cycle times and up to 15 times higher idle times than TT-PS.

The detailed comparison of the three algorithms is presented in Table 2, where each row stands for a separate problem instance. Instance names beginning with W and WF refer to welding without fixture and with fixture, respectively. Column n contains the number of stitches, while *min. accessibility* and *avg. accessibility* present the minimum and average accessibility ratio, i.e., the ratio of CFAV and TAV, measured over the different stitches in percent. For each algorithm, columns *cycle1* and *cycle2* contain the cycle time of the rough-cut path and the collision-free path, respectively. The best cycle times are denoted by bold font for each instance. Columns *run* contain the run time of the algorithm in seconds. It is noted that for RMV and RMV*, the TSP solver terminated with a locally optimal sequence in less then 1 second, hence, *run* is practically the time required for collision avoidance. In contrast, TS-PP was run for 120 seconds on each instance, plus the time of collision avoidance.

The results show a notable difference among the instances depending on stitch accessibility. For the WF instances, accessibility was very poor (minimum accessibility around 10%, average accessibility of 60-70%). For the W instances, collision avoidance was run with the workpiece geometry only, resulting in 24-50% min. and around 90% average accessibility. The reason of poor accessibility with fixture was twofold. First, the car door was originally designed for spot welding, and the stitch layout was received by replacing the spots by RLW stitches, with minor modifications; in fact, ca. 20-40% less stitches could ensure sufficient stiffness. On the other hand, the key design objective for the experimental fixture was to achieve perfect gap control, while the general design guideline that the stitch accessibility volumes must be kept clear was ignored. It is noted that several instances had to be pre-processed to eliminate stitches that are completely inaccessible, since otherwise the path planning problem would have no feasible solution. After all, we expect that for a car door in production, stitch accessibility and the complexity of collision avoidance would be somewhere between those experienced for the W and the WF instances.

Regarding algorithm performance, TS-PP reduced cycle times drastically compared to RMV. The reduction was on average 63% on the rough-cut path, and 61% on the collision-free path. This was mostly due to the joint consideration of the TCP and the scanner movement, instead of optimizing the TCP path only.

TS-PP also outperformed RMV* regarding the cycle time of the rough-cut path on every instance, by computing up to 6.1%, on average 2.9% more efficient paths. However, this did not automatically translate to improvement on the collision-free path on each individual instance. The perturbation of the rough-cut paths by collision avoidance resulted in a situation where TS-PP computed better collision-free paths on 11 out of 15 instances, by up to 4.3%. However, RMV* outperformed TS-PP on 4 of the 15 instances, by 2.1-4.9% on the different instances. This occurred typically for the WF instances with the worst accessibility. Beyond the random perturbation caused by the modifications to the rough-cut path, a possible explanation of this phenomenon comes from the different underlying assumptions made by the algorithms for sequencing. Implicitly, RMV* assumes that each stitch is welded from the mid-point of the technological access volume, whereas TS-PP assumes that the complete technological access volume can be used. In these problematic instances, the assumption of RMV* appears to be closer to reality. Initial experiments on sequencing using reduced TAVs confirm this hypothesis, and with an appropriate choice of parameters, the method resulted in TS-PP outperforming RMV* an all instance, but an elaborate heuristic is subject to future work.

The average computation time was 165 seconds and 119 seconds for RMV and RMV*. TS-PP required 241 seconds on average, due to the higher computation time of sequencing. Half of the computation time was taken by sequencing and rough-cut path planning, while the other half by collision-free path planning. Still, these response times comply with industrial expectations, and enable the use of the algorithms in a decision support tool in an iterative design and planning process.

Conclusions

This paper introduced a new collision-free path planning algorithm for RLW. The algorithm departs from a task sequence and a potentially colliding rough-cut path, and alters this path to achieve a collision-free path with minimal

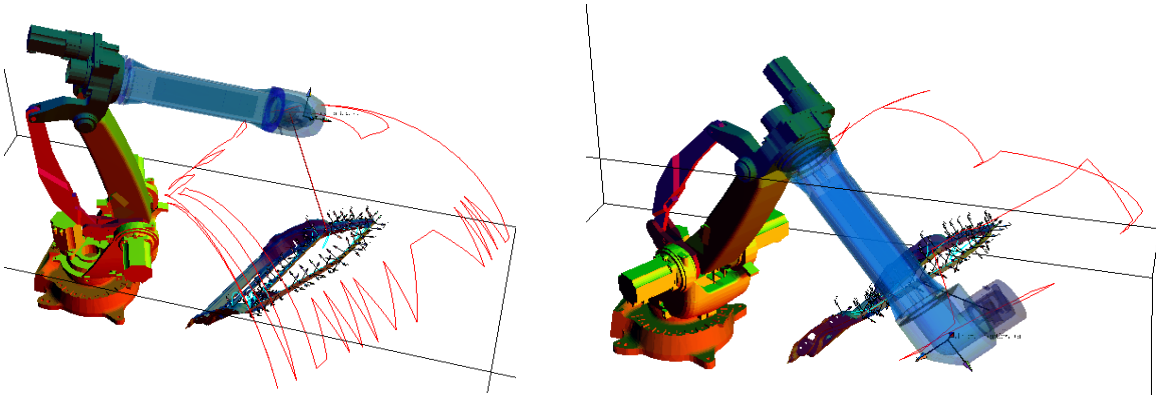

Figure 9: Comparison of the paths computed by the RMV (left) and the proposed TS-PP (right) methods. TS-PP focuses on the scanner path, and geometrical and technological parameters already at the time of task sequencing, which results in shorter scanner path and reduced cycle time.

		Accessibility		RMV			RMV*			TS-PP		
	n	min.	avg.	cycle1	cycle2	run	cycle1	cycle2	run	cycle1	cycle2	run
W1	28	47.32	91.63	30.05	30.53	22	14.01	14.01	7	**13.69**	**13.69**	128
W2	34	47.32	95.16	35.50	35.50	2	15.93	15.93	2	**15.48**	**15.49**	135
W3	62	49.29	93.61	76.39	76.64	11	26.91	26.91	2	**26.11**	**26.11**	122
W4	44	34.38	87.21	56.33	57.23	19	19.55	19.84	8	**18.36**	**18.98**	146
W5	71	24.46	90.76	78.64	78.64	3	30.29	30.29	3	**29.85**	**29.85**	123
W6	67	24.46	90.84	67.70	67.70	3	28.50	28.50	3	**27.75**	**27.75**	123
WF1	28	10.97	64.21	30.05	31.26	229	14.01	15.04	113	**13.69**	**14.69**	299
WF2	34	14.63	69.49	35.50	35.86	286	15.93	16.92	192	**15.48**	**16.42**	337
WF3	62	11.14	68.49	76.39	78.42	294	26.91	**27.51**	219	**26.11**	28.08	353
WF4	44	10.89	58.81	56.33	58.23	163	19.55	21.16	196	**18.37**	**21.02**	283
WF5	64	9.79	65.03	75.37	77.18	270	26.15	**27.58**	249	**26.10**	28.93	448
WF6	63	9.79	63.55	74.92	76.40	399	25.81	**27.25**	365	**25.07**	27.91	404
WF7	63	6.90	60.98	74.92	76.59	504	25.81	**27.38**	334	**25.07**	27.95	421
Avg.	51	21.04	72.18	59.08	60.01	170	22.26	22.95	130	**21.63**	**22.84**	256

Table 2: Comparison of the RMV, RMV*, and the proposed TS-PP algorithms.

cycle time by iterating shortest path algorithms and distance queries on a mesh model representation of the involved moving objects. Extensive computational experiments have shown that the proposed algorithms are efficient in solving real industrial problems originating from the automotive industry.

Nevertheless, the results achieved permit drawing conclusions in a wider context as well. Most importantly, it has been shown that in RLW, and in general, for machining technologies where relatively slow robot motion is coupled with quick movements of the tool, optimization must jointly consider the robot path and the tool path, instead of focusing solely on the tool path. For the car door designs considered in our experiments, this resulted in an enormous reduction of the cycle times, by 63% for on average.

Second, while tool positions are well defined for the effective tasks, e.g., stitch positions in RLW, one has a significant degree of freedom in choosing the corresponding robot path. On the one hand, this freedom opens new opportunities for optimization, but on the other hand, it presents a serious computational challenge, and an efficient combination of combinatorial optimization and geometric reasoning is required for tackling it. While most earlier contributions applied a sampling strategy to solve sequencing and path planning over a finite set of pre-defined discrete points, we proposed algorithms for planning in the continuous space, using efficient geometric computation routines.

Our current research focuses on improving the stitch sequence and the rough-cut path on instances with poor accessibility, by heuristics that adjust the technological access volumes to the real, collision-free access volumes. Furthermore, the verification and thorough evaluation of the developed off-line programming toolbox in physical experiments is underway.

Acknowledgements

The author thanks József Váncza and Gábor Erdős for the helpful discussions. This work has been supported by EU FP7 grant RLW Navigator No. 285051 and the NFÜ grant ED-13-2-2013-0002.

References

Alatartsev, S.; Augustine, M.; and Ortmeier, F. 2013. Constricting insertion heuristic for traveling salesman problem with neighborhoods. In *Proc. of the 23rd International Conference on Automated Planning and Scheduling (ICAPS-2013)*, 2–10.

Alatartsev, S.; Mersheeva, V.; Augustine, M.; and Ortmeier, F. 2013. On optimizing a sequence of robotic tasks. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS2013)*, 217–223.

Castelino, K.; D'Souza, R.; and Wright, P. K. 2002. Toolpath optimization for minimizing airtime during machining. *Journal of Manufacturing Systems* 22(3):173–180.

Dewil, R.; Vansteenwegen, P.; and Cattrysse, D. 2014. Construction heuristics for generating tool paths for laser cutters. *International Journal of Production Research* in print.

Erdős, G.; Kemény, Z.; Kovács, A.; and Váncza, J. 2013. Planning of remote laser welding processes. *Procedia CIRP* 7:222–227.

Ferguson, D., and Stentz, A. 2006. Using interpolation to improve path planning: The field D* algorithm. *Journal of Field Robotics* 23(2):79–101.

Geraerts, R., and Overmars, M. H. 2002. A comparative study of probabilistic roadmap planners. In *Proc. Workshop on the Algorithmic Foundations of Robotics*, 43–57.

Hatwig, J.; Minnerup, P.; Zaeh, M. F.; and Reinhart, G. 2012. An automated path planning system for a robot with a laser scanner for remote laser cutting and welding. In *2012 IEEE International Conference on Mechatronics and Automation (ICMA)*, 1323–1328.

Hatwig, J.; Reinhart, G.; and Zaeh, M. F. 2010. Automated task planning for industrial robots and laser scanners for remote laser beam welding and cutting. *Production Engineering* 4(4):327–332.

Hudson, T. C.; Lin, M. C.; Cohen, J.; Gottschalk, S.; and Manocha, D. 1997. V-collide: Accelerated collision detection for vrml. In *VRML 97: Second Symposium on the Virtual Reality Modeling Language*, 119–125.

Kaelbling, L., and Lozano-Perez, T. 2011. Hierarchical task and motion planning in the now. In *2011 IEEE International Conference on Robotics and Automation (ICRA)*, 1470–1477.

Kavraki, L. E.; Svestka, P.; Latombe, J.-C.; and Overmars, M. H. 1996. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation* 12(4):566–580.

Koenig, S., and Likhachev, M. 2005. Fast replanning for navigation in unknown terrain. *IEEE Transactions on Robotics* 21(3):354–363.

Kolakowska, E.; Smith, S. F.; and Kristiansen, M. 2014. Constraint optimization model of a scheduling problem for a robotic arm in automatic systems. *Robotics and Autonomous Systems* 62(2):267–280.

Kovács, A. 2013. Task sequencing for remote laser welding in the automotive industry. In *Proceedings of the 23rd International Conference on Automated Planning and Scheduling (ICAPS-2013)*, 457–461.

Kucuk, S., and Bingul, Z. 2006. Robot kinematics: Forward and inverse kinematics. In Cubero, S., ed., *Industrial Robotics: Theory, Modelling and Control*. Pro Literatur Verlag. 117–148.

Kuffner, J. J., and LaValle, S. M. 2000. RRT-connect: An efficient approach to single-query path planning. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'00)*, 995–1001.

Larsen, E.; Gottschalk, S.; Lin, M. C.; and Manocha, D. 2000. Fast proximity queries with swept sphere volumes. In *Proc. IEEE Int. Conf. Robot. Autom.*, 3719–3726.

Likhachev, M.; Gordon, G.; and Thrun, S. 2003. ARA*: Anytime A* with provable bounds on sub-optimality. In *Advances in Neural Information Processing Systems (NIPS)*.

Park, H.-S., and Choi, H.-W. 2010. Development of digital laser welding system for car side panels. In Na, X., ed., *Laser Welding*. InTech. 181–192.

Park, C.; Pan, J.; Lin, M.; and Manocha, D. 2013. Realtime gpu-based motion planning for task execution in dynamic environments. In *Proceedings of the 1st Workshop on Planning and Robotics (PlanRob 2013)*, 60–63.

Reinhart, G.; Munzert, U.; and Vogl, W. 2008. A programming system for robot-based remote-laser-welding with conventional optics. *CIRP Annals – Manufacturing Technology* 57(1):37–40.

Saha, M.; Sánchez-Ante, G.; Roughgarden, T.; and Latombe, J.-C. 2006. Planning tours of robotic arms among partitioned goals. *International Journal of Robotics Research* 25(3):207–223.

Shibata, K. 2008. Recent automotive applications of laser processing in Japan. *The Review of Laser Engineering* 36:1188–1191.

Srivastava, S.; Fang, E.; Riano, L.; Chitnis, R.; Russell, S.; and Abbeel, P. 2014. Combined task and motion planning through an extensible planner-independent interface layer. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*.

Stentz, A. 1995. The focussed D* algorithm for real-time replanning. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'95)*, 1652–1659.

Trenkel, S.; Weller, R.; and Zachmann, G. 2007. A benchmarking suite for static collision detection algorithms. In *International Conference in Central Europe on computer graphics, visualization and computer vision (WSCG)*.

Tsoukantas, G.; Salonitis, K.; Stournaras, A.; Stavropoulos, P.; and Chryssolouris, G. 2007. On optical design limitations of generalized two-mirror remote beam delivery laser systems: the case of remote welding. *The International Journal of Advanced Manufacturing Technology* 32(9–10):932–941.